

TUNABLE NARROW-BAND FILTER INCLUDING
SIGMA-DELTA MODULATOR

Christopher H. Dick
Frederic J. Harris

CROSS-REFERENCE TO RELATED APPLICATION

This is a continuation-in-part of U.S. Patent Application
Serial No. 09/394,123 filed 9/10/1999 entitled "Narrow Band
Filter Including Sigma-Delta Modulator Implemented in a
Programmable Logic Device", incorporated herein by reference.

INTRODUCTIONBACKGROUND

While $\Sigma\Delta$ techniques are applied widely in analog
conversion sub-systems, both analog-to-digital (ADC) and
digital-to-analog (DAC) converters, these methods have
enjoyed much less exposure in the broader application
domain, where flexible and configurable solutions,
traditionally supplied via a software DSP (soft-DSP), are
required. And this limited level of exposure is easy to
understand. Most, if not all, of the efficiencies and
optimizations afforded by $\Sigma\Delta$ are hardware oriented and so
cannot be capitalized on in the fixed precision pre-defined
datapath found in a soft-DSP processor. This limitation, of
course, does not exist in a field programmable gate array
(FPGA) DSP solution. With FPGAs the designer has complete
control of the silicon to implement any desired datapath and
employ optimal word precisions in the system with the
objective of producing a design that satisfies the
specifications in the most economically sensitive manner.

RED-LINED VERSION

1 While implementation of a digital $\Sigma\Delta$ ASIC (application-
2 specific integrated circuit) is of course possible, economic
3 constraints make the implementation of such a building block
4 that would provide the flexibility, and be generic enough to
5 cover a broad market cross-section, impractical. FPGA-based
6 hardware provides a solution to this problem. FPGAs are off-
7 the-shelf commodity items that provide a silicon feature set
8 ideal for constructing high-performance DSP systems. These
9 devices maintain the flexibility of software-based
10 solutions, while providing levels of performance that match,
11 and often exceed, ASIC solutions.

12 There is a rich and expanding body of literature
13 devoted to the efficient and effective implementation of
14 digital signal processors using FPGA based hardware. More
15 often than not, the most successful of these techniques
16 involves a paradigm shift away from the methods that provide
17 good solutions in software programmable DSP systems.

18 ~~This paper reports on the rich set of design~~
19 ~~opportunities that are available to the signal processing~~
20 ~~system designer through innovative combinations of $\Sigma\Delta$~~
21 ~~techniques and FPGA signal processing hardware. The~~
22 ~~applications considered include narrow band filters, both~~
23 ~~single rate and multi rate, DC canceler, and $\Sigma\Delta$ hybrid~~
24 ~~digital analog control loops for simplifying carrier~~
25 ~~recovery, timing recovery, and AGC (automatic gain control)~~
26 ~~loops in a digital communication receiver.~~

27 ~~This application is organized as follows: section 2~~
28 ~~presents a brief overview of FPGA architecture. In section 3~~
29 ~~a simple single loop base band $\Sigma\Delta$ modulator is introduced.~~
30 ~~The structure is an extended to a normal architecture that~~
31 ~~permits center frequency tuning, as well as a method for~~

RED-LINED VERSION

1 ~~working with the system degrees of freedom to trade off~~
2 ~~modulator bandwidth with dynamic range. The tunable $\Sigma\Delta$ is~~
3 ~~then utilized for implementing area efficient FPGA FIR~~
4 ~~filters. The process for computing the modulator~~
5 ~~coefficients for low pass, bandpass, and high pass designs~~
6 ~~is described. In section 4, a new $\Sigma\Delta$ architecture is~~
7 ~~described that provides a very simple method for tuning~~
8 ~~using only a single coefficient. In any fixed point data~~
9 ~~path, careful consideration must be given to the DC aspects~~
10 ~~of the design. For example, the introduction of a DC~~
11 ~~complement to two step alteration between the stages of a~~
12 ~~multi stage, multi rate filter can be problematic, causing~~
13 ~~arithmetic saturation or increasing the bit error rates in~~
14 ~~additional receiver. Section 5 describes a unique $\Sigma\Delta$~~
15 ~~approach to building a DC canceler. In section 6, $\Sigma\Delta$~~
16 ~~methods are described for simplifying the implementation of~~
17 ~~hybrid digital analog control loops in a system such as a~~
18 ~~software defined radio. In section 7 some comments on the~~
19 ~~industrial implications of the techniques considered in the~~
20 ~~application are presented. Finally, some conclusions are~~
21 ~~drawn in section 8.~~

22 Semiconductor vendors, such as Xilinx, Altera, Atmel,
23 and AT&T, provide a range of FPGAs. The architectural
24 approaches are as diverse as there are manufacturers, but
25 some generalizations can be made. Most of the devices are
26 basically organized as an array of logic elements and
27 programmable routing resources used to provide the
28 connectivity between the logic elements, FPGA I/O pins and
29 other resources, such as on-chip memory. The structure and
30 complexity of the logic elements, as well as the
31 organization and functionality supported by the

RED-LINED VERSION

1 interconnection hierarchy, distinguish the devices. Other
2 device features, such as block memory and delay locked loop
3 technology, are also significant factors that influence the
4 complexity and performance of an algorithm that is
5 implemented using FPGAs.

6 A logic element usually consists of one or more RAM
7 (random access memory) n-input look-up tables, where n is
8 between ~~three~~3 and 6, in one to several flip-flops. There
9 may also be additional hardware support in each element to
10 enable high-speed arithmetic operations. This generic FPGA
11 architecture is shown in Figure 1. Also illustrated in the
12 Figure (as wide lines) are several connections between logic
13 elements and the device input/output (I/O) ports.

14 Application-specific circuitry is supported in the device by
15 downloading a bit stream into SRAM (static random access
16 memory) based configuration memory. This personalization
17 database defines the functionality of the logic elements, as
18 well as the internal routing. Different applications are
19 supported on the same FPGA hardware platform by configuring
20 the FPGA(s) with appropriate bit streams. As a specific
21 example, consider the Xilinx Virtex™ series of FPGAs. The
22 logic elements, called slices, essentially ~~consistent~~
23 ~~te~~**consist of two** four-input look-up tables (LUTs), two flip-
24 flops, several multiplexors and some additional silicon
25 support that allows the efficient implementation of carry-
26 chains for building high-speed ~~matters~~**adders**, subtracters,
27 and shift registers. Two slices form a configurable logic
28 block (CLB) as shown in Figure 2. The CLB is the basic tile
29 **that** is used to build the logic matrix. Some FPGAs, like the
30 Xilinx Virtex families, supplying on-chip block RAM. Figure
31 3 shows the CLB matrix that defines a Virtex FPGA. Current

RED-LINED VERSION

1 generation Virtex silicon provides a family of devices
2 offering 768 to 12,288 logic slices, and from 8 to 32
3 variable form factor block memories.

4 Xilinx XC4000 and Virtex devices also ~~all-out~~**allow** the
5 designer to use the logic element LUTs as ~~memory~~
6 ~~either~~**memory, either** ROM or RAM. Constructing memory with
7 this distributed memory approach can yield access bandwidths
8 in many ~~tens~~ ~~per second~~ **gigabytes** per second range.

9 Typical clock frequencies for current generation
10 devices are in the multiple ~~tenants~~ of megahertz (100 to
11 200) range.

12 In contrast to the logic slice architecture employed in
13 Xilinx Virtex devices, a logic block architecture employed
14 in the Atmel AT40K FPGA is shown in Figure 4. Like the
15 Xilinx device, combinational logic is realized using look-up
16 tables. In this case, ~~two~~ three-input LUTs and a single
17 flip-flop are available in each logic cell. The pass gates
18 in a cell form part of the signal routing network and are
19 used for connecting signals to the multiple horizontal and
20 vertical bus ~~planes~~**planes**. In addition to the orthogonal
21 routing resources, indicated as N, S, E and W in Figure 4, a
22 diagonal group of interconnects (NW, NE, SE, and SW),
23 associated with each cell x output, are available to provide
24 efficient connections to neighboring cell's x bus inputs.

25 The objective of the FPGA/DSP architect is to formulate
26 algorithmic solutions for applications that best utilize
27 FPGA resources to achieve the required functionality. This
28 is a three-dimensional optimization problem in power,
29 complexity, and bandwidth. The remainder of this application
30 describes some novel FPGA solutions to several signal
31 processing problems. The results are important in ~~an~~

RED-LINED VERSION

1 industrial context because they enable either smaller, and
2 hence more economic, solutions to important problems, or
3 allow more arithmetic compute power to be realized with a
4 given area of silicon.

5
6
7 ~~$\Sigma\Delta$ MODULATORS, FIR FILTERS AND FPGAS~~

8 ~~$\Sigma\Delta$ based DSP is employed to generate FPGA hardware~~
9 ~~implementations of narrow band filters, a DC canceller, and~~
10 ~~hybrid digital analog control loops for a software defined~~
11 ~~radio architecture.~~

12
13 ~~This section describes a method employing sigma delta~~
14 ~~modulation ($\Sigma\Delta$) techniques for implementing area efficient~~
15 ~~finite impulse response (FIR) filters using FPGA hardware.~~
16 ~~Before treating the FPCA filter design, a brief review of $\Sigma\Delta$~~
17 ~~modulation encoding is presented.~~

18
19 ~~$\Sigma\Delta$ Modulation~~

20 Sigma-Delta modulation is a source coding technique
21 most prominently employed in analog-to-digital and digital-
22 to-analog converters. In this context, hybrid analog and
23 digital circuits are used in the realization. Figure 5 shows
24 a single-loop $\Sigma\Delta$ modulator. Provided the input signal is
25 busy enough, the linearized discrete time model of Figure 6
26 can be used to illustrate the principle. In Figure 6, the 1-
27 bit quantizer is modeled by an additive white noise source
28 with variance $\sigma_e^2 = \Delta^2/12$, where Δ represents the quantization
29 interval. The ~~z-transform~~**z-transform** of the system is

30

1 Equations 1 and 2:

$$\begin{aligned} Y(z) &= \frac{H(z)}{1 + H(z)} X(z) + \frac{1}{1 + H(z)} Q(z) \\ &= H_s(z) X(z) + H_n(z) Q(z) \end{aligned}$$

2 where

3

4 Equation 3:

$$H(z) = \frac{1}{z - 1}$$

5

6 which is the transfer function of delay and an ideal
7 integrator, and $H_s(z)$ and $H_n(z)$ are the signal and noise
8 transfer functions (NTF) respectively. In a good $\Sigma\Delta$
9 modulator, $H_n(\omega)$ will have a flat frequency response in the
10 interval $|f| \leq B$. In contrast, $H_n(\omega)$ will have a high
11 attenuation in the frequency band $|f| \leq B$ and a "don't care"
12 region in the interval $B < |f| < f_s/2$. For the single loop $\Sigma\Delta$
13 in Figure 6, $H_s(z) = z^{-1}$ and $H_n(z) = 1 - z^{-1}$. Thus the input
14 signal is not distorted in any way by the network and simply
15 experiences a pure delay from input to output. The
16 performance of the system is determined by the noise
17 transfer function $H_n(z)$, which is given by

18

19 Equation 4:

$$|H_n(f)| = 4 \left| \sin \frac{\pi f}{f_s} \right|$$

20

21 and is shown in Figure 7. The in-band quantization noise
22 variance is

23

1 Equation 5:

$$\sigma_n^2 = \int_{-B}^{+B} |H_n(f)|^2 S_q(f) df$$

2

3 where $S_q(f) = \sigma_q^2 / f_s$ is the power spectral density of the
4 quantization noise. Observe that for a non-shaped noise (or
5 white) spectrum, increasing the sampling rate by a factor of
6 2, while keeping the bandwidth B fixed, reduces the
7 quantization noise by 3 dB. For a first order $\Sigma\Delta$ it can be
8 shown that

9

10 Equation 6:

$$\sigma_n^2 \approx \frac{1}{3} \pi^2 \sigma_q^2 \left(\frac{2B}{f_s} \right)^3$$

11

12 for $f_s \gg 2B$. Under these conditions doubling the sampling
13 frequency reduces the noise power by 9 dB, of which 3 dB is
14 due to the reduction in $S_q(f)$ and a further 6 dB is due to
15 the filter characteristic $H_n(f)$. The noise power is reduced
16 by increasing the sampling rate to spread the quantization
17 noise over a large bandwidth and then by shaping the power
18 spectrum using an appropriate filter.

19

20 Reduced Complexity Filters Using $\Sigma\Delta$ Modulation Techniques

21 $\Sigma\Delta$ techniques can be employed for realizing area
22 efficient narrowband filters in FPGAs. These filters are
23 utilized in many applications. For example, narrow-band
24 communication receivers, multi-channel RF surveillance
25 systems and for solving some spectrum management problems.

1 A uniform quantizer operating at the Nyquist rate is
2 the standard solution to the problem of representing data
3 within a specified dynamic range. Each additional bit of
4 resolution in the quantizer provides an increase in dynamic
5 range of approximately 6dB. A signal with 60dB of dynamic
6 range requires 10 bits, while 16 bits can represent data
7 with a dynamic range of 96dB.

8 While the required dynamic range of a system fixes the
9 number of bits required to represent the data, it also
10 affects the expense of subsequent arithmetic operations, in
11 particular multiplications. In any hardware implementation,
12 and of course this includes FPGA based DSP processors, there
13 are strong economic imperatives to minimize the number and
14 complexity of the arithmetic components employed in the
15 datapath. An embodiment of the invention employs noise-
16 shaping techniques to reduce the precision of the input data
17 samples to minimize the complexity of the multiply-
18 accumulate (MAC) units in the filter. The net result is a
19 reduction in the amount of FPGA logic resources required to
20 realize the specified filter.

21 Consider the structure shown in Figure 8. Instead of
22 applying the quantized data $x(n)$ from the analog-to-digital
23 converter directly to the filter, data $x(n)$ is pre-processed
24 by a $\Sigma\Delta$ modulator. The re-quantized input samples $\hat{x}(n)$ are
25 represented using fewer bits per sample, so permitting the
26 subsequent filter $H(z)$ to employ reduced precision
27 multipliers in the mechanization. The filter coefficients
28 are still kept to a high precision.

29 The $\Sigma\Delta$ data re-quantizer is based on a single loop
30 error feedback sigma-delta modulator shown in Figure 9. In
31 this configuration, the difference between the quantizer

1 input and output sample is a measure of the quantization
2 error, which is fed back and combined with the next input
3 sample. The error-feedback sigma-delta modulator operates on
4 a highly oversampled input and uses the unit delay z^{-1} as a
5 predictor. With this basic error-feedback modulator, only a
6 small fraction of the bandwidth can be occupied by the
7 required signal. In addition, the circuit only operates at
8 baseband. A larger fraction of the Nyquist bandwidth can be
9 made available and the modulator can be tuned if a more
10 sophisticated error predictor is employed. This requires
11 replacing the unit delay with a prediction filter $P(z)$. This
12 generalized modulator is shown in Figure 10.

13 The operation of the re-quantizer can be understood by
14 considering the transform domain description of the circuit.
15 This is expressed as

16

17 Equation 7:

$$\hat{X}(z) = X(z) + Q(z)(1 - P(z)z^{-1})$$

18

19 where $Q(z)$ is the z -transform of the equivalent noise source
20 added by the quantizer $q(\cdot)$, $P(z)$ is the transfer function of
21 the error predictor filter, and $X(z)$ and $\hat{X}(z)$ are the
22 transforms of the system input and output respectively. $P(z)$
23 is designed to have unity gain and leading phase shift in
24 the bandwidth of interest. Within the design bandwidth, the
25 term $Q(z)(1 - P(z)z^{-1}) = 0$ and so $X(z) = \hat{X}(z)$. By designing $P(z)$
26 to be commensurate with the system passband specifications,
27 the in-band spectrum of the re-quantizer output will ideally
28 be the same as the corresponding spectral region of the
29 input signal.

1 To illustrate the operation of the system consider the
2 task of recovering a signal that occupies 10% of the
3 available bandwidth and is centered at a normalized
4 frequency of 0.3Hz. The stopband requirement is to provide
5 60 dB of attenuation. Figure 11A shows the input test
6 signal. It comprises an in-band component and two out-of-
7 band tones that are to be rejected. Figure 11B is a
8 frequency domain plot of the signal after it has been re-
9 quantized to 4 bits of precision by a $\Sigma\Delta$ modulator employing
10 an 8th order predictor in the feedback path. Notice that the
11 60dB dynamic range requirement is supported in the bandwidth
12 of interest, but that the out-of-band SNR has been
13 compromised. This is of course acceptable, since the
14 subsequent filtering operation will provide the necessary
15 rejection. A 160-tap filter $H(z)$ satisfies the problem
16 specifications. The frequency response of $H(z)$ using 12-bit
17 filter coefficients is shown in Figure 11C. Finally, $H(z)$ is
18 applied to the reduced sample precision data stream $\hat{X}(z)$ to
19 produce the spectrum shown in Figure 11D. Observe that the
20 desired tone has been recovered, the two out-of-band
21 components have been rejected, and that the in-band dynamic
22 range meets the 60 dB requirement.

23

24 Prediction Filter Design

25 The design of the error predictor filter is a signal
26 estimation problem. The optimum predictor is designed from a
27 statistical viewpoint. The optimization criterion is based
28 on the minimization of the mean-squared error. As a
29 consequence, only the second-order statistics
30 (autocorrelation function) of a stationary process are
31 required in the determination of the filter. The error

RED-LINED VERSION

1 predictor filter is designed to predict samples of a band-
 2 limited white noise process $N_{xx}(\omega)$ shown in Figure 12.

3 $N_{xx}(\omega)$ is defined as:

4

5 Equation 8:

$$6 \quad N_{xx}(\omega) = \begin{cases} 1 & -\theta \leq \omega \leq \theta \\ 0 & \text{otherwise} \end{cases}$$

7

8 and related to the autocorrelation sequence $r_{xx}(m)$ by
 9 discrete-time Fourier transform (DTFT).

10

11 Equation 9:

$$12 \quad N_{xx}(\omega) = \sum_{n=-\infty}^{\infty} r_{xx}(n) e^{-j\omega n}$$

14

15 The autocorrelation function $r_{xx}(n)$ is found by taking the
 16 inverse DTFT of the equation immediately above.

17

18 Equation 10:

$$19 \quad r_{xx}(n) = \frac{1}{2\pi} \int_{-\pi}^{\pi} N_{xx}(\omega) e^{-j\omega n} d\omega$$

20

21

22 $N_{xx}(\omega)$ is non-zero only in the interval $-\theta \leq \omega \leq \theta$ giving $r_{xx}(n)$
 23 as:

24

25 Equation 11:

$$26 \quad r_{xx}(n) = \frac{\theta}{\pi} \text{sinc}(\theta n)$$

27

28 So the autocorrelation function corresponding to a band-
 29 limited white noise power spectrum is a sinc function. Samples
 30 of this function are used to construct an autocorrelation

RED-LINED VERSION

1 matrix which is used in the solution of the normal equations
 2 to find the required coefficients. Leaving out the scaling
 3 factor in the immediately above equation, the required
 4 autocorrelation function $r_{xx}(n)$, truncated to p samples, is
 5 defined as:

6

7 Equation 12:

8

$$9 \quad r_{xx} = \frac{\sin(n\theta)}{n\theta} \quad n = 0, \dots, p-1$$

10

11

12 The normal equations are defined as:

13

14 Equation 13:

15

$$16 \quad r_{xx}(m) = \sum_{k=1}^p a(k) r_{xx}(m-k) \quad m = 1, 2, \dots, p$$

17

18

19 This system of equations can be compactly written in
 20 matrix form by first defining several matrices.

21 To design a p -tap error predictor filter first compute a
 22 sinc function consisting of $p+1$ samples and construct the
 23 autocorrelation matrix R_{xx} as:

24

25 Equation 14:

26

$$27 \quad R_{xx} = \begin{bmatrix} r_{xx}(0) & r_{xx}(1) & \dots & r_{xx}(p-1) \\ r_{xx}(1) & r_{xx}(0) & \dots & r_{xx}(p-2) \\ \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot \\ r_{xx}(p-1) & r_{xx}(p-2) & \dots & r_{xx}(0) \end{bmatrix}$$

30

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30

Next, define a filter coefficient row-vector A as:

Equation 15:

$$A = [a(0), a(1), \dots, a(p-1)]$$

where $a(i), i=0, \dots, p-1$, are the predictor filter coefficients. Let the row-vector R'_{xx} be defined as:

Equation 16:

$$R'_{xx} = [r_{xx}(1), r_{xx}(2), \dots, r_{xx}(p)]$$

The matrix equivalent of equation 13 is:

Equation 17:

$$R_{xx} A^T = (R'_{xx})^T$$

The filter coefficients are therefore given as:

Equation 18:

$$A^T = R_{xx}^{-1} (R'_{xx})^T$$

For the case in-hand, the solution of equation 18 is an ill-conditioned problem. To arrive at a solution for A , a small constant ϵ is added to the elements along the diagonal of the autocorrelation matrix R_{xx} in order to raise its condition number. The actual autocorrelation matrix used to solve for the predictor filter coefficients is:

Equation 19:

$$R_{xx} = \begin{bmatrix} r_{xx}(0) + \varepsilon & r_{xx}(1) & \dots & r_{xx}(p-1) \\ r_{xx}(1) & r_{xx}(0) + \varepsilon & \dots & r_{xx}(M-2) \\ \vdots & \vdots & \ddots & \vdots \\ r_{xx}(p-1) & r_{xx}(p-2) & \dots & r_{xx}(0) + \varepsilon \end{bmatrix}$$

Bandpass Predictor Filter

The previous section described the design of a lowpass predictor. In this section, bandpass processes are considered.

A bandpass predictor filter is designed by modulating a lowpass prototype sinc function to the required center frequency θ_0 . The bandpass predictor coefficient $h_{BP}(n)$ is obtained from the prototype lowpass sinc function $h_{LP}(n)$ as:

Equation 20:

$$\text{sinc}_{BP}(n) = \text{sinc}_{LP}(n) \cos(\theta_0(n-k)) \quad n = 0, \dots, 2p$$

$$\text{where } k = \left\lceil \frac{2p+1}{2} \right\rceil.$$

Highpass Predictor Filter

A highpass predictor filter is designed by highpass modulating a lowpass prototype sinc function to the required corner frequency θ_c . The highpass predictor coefficients $h_{HP}(n)$ are obtained from the prototype lowpass sinc function $h_{LP}(n)$ as:

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31

Equation 21:

$$\text{sinc}_{HP}(n) = \text{sinc}_{LP}(n) (-1)^{n-k} \quad n = 0, \dots, 2p$$

$\Sigma\Delta$ Modulator FPGA Implementation

The most challenging aspect of implementing the data modulator is producing an efficient implementation for the prediction filter $P(z)$. The desire to support high-sample rates, and the requirement of zero latency for $P(z)$, will preclude bit-serial methods from this problem. In addition, for the sake of area efficiency, parallel multipliers that exploit one time-invariant input operand (the filter coefficients) will be used, rather than general variable-variable multipliers. The constant-coefficient multiplier (KCM) is based on a multi-bit inspection version of Booth's algorithm. Partitioning the input variable into 4-bit nibbles is a convenient selection for the Xilinx Virtex function generators (FG). Each FG has 4 inputs and can be used for combinatorial logic or as application RAM/ROM. Each logic slice in the Virtex logic fabric comprises 2 FGs, and so can accommodate a 16×2 memory slice. Using the rule of thumb that each bit of filter coefficient precision contributes 5 dB to the sidelobe behavior, 12-bit precision is used for $P(z)$. 12-bit precision will also be employed for the input samples. There are 3 4-bit nibbles in each input sample. Concurrently, each nibble addresses independent 16×16 lookup tables (LUTs). The bit growth incorporated here allows for worst case filter coefficient scaling in $P(z)$. No pipeline stages are permitted in the multipliers because of $P(z)$'s location in the feedback path of the modulator.

1 It is convenient to use the transposed FIR filter for
2 constructing the predictor. This allows the adders and delay
3 elements in the structure to occupy a single slice. 64
4 slices are required to build the accumulate-delay path. The
5 FPGA logic requirements for $P(z)$, using a 9-tap predictor,
6 is $\Gamma(P(z)) = 9 \times 40 + 64 = 424$ CLBs. A small amount of
7 additional logic is required to complete the entire $\Sigma\Delta$
8 modulator. The final slice count is 450. The entire
9 modulator comfortably operates with a 113 MHz clock. This
10 clock frequency defines the system sample rate, so the
11 architecture can support a throughput of 113 MSamples per
12 second. The critical path through this part of the design is
13 related to the exclusion of pipelining in the multipliers.

14

15 Reduced Complexity FIR Mechanization

16 Now that the input signal is available as a reduced
17 precision sample stream, filtering can be performed using
18 area-optimized hardware. For the reasons discussed above, 4-
19 bit data samples are a convenient match for Virtex devices.
20 Figure 13 shows the structure of the reduced complexity FIR
21 filter. The coded samples $\hat{x}(n)$ are presented to the address
22 inputs of N coefficient LUTs. In accordance with the
23 modulated data stream precision, each LUT stores the 16
24 possible scaled coefficient values for one tap as shown in
25 Figure 14. An N-tap filter requires N such elements. The
26 outputs of the minimized multipliers are combined with an
27 add-delay datapath to produce the final result. The logic
28 requirement for the filter is $\Gamma(H(z)) = N\Gamma(MUL) + (N-1)\Gamma(ADD_z^{-1})$
29 where $\Gamma(MUL)$ and $\Gamma(ADD_z^{-1})$ are the FPGA area cost functions
30 for a KCM multiplier and an add-delay datapath component

1 respectively.

2 Using full-precision input samples without any $\Sigma\Delta$
3 encoding, each KCM would occupy 40 slices. The total cost of
4 a direct implementation of $H(z)$ is 7672 slices. The reduced
5 precision KCMs used to process the encoded data each consume
6 only 8 slices. Including the sigma-delta modulator the slice
7 count is 3002 for the $\Sigma\Delta$ approach. So the data re-
8 quantization approach consumes only 39% of the logic
9 resources of a direct implementation.

10

11 $\Sigma\Delta$ Decimators

12 The procedure for re-quantizing the source data can
13 also be used effectively in an m:1 decimation filter. An
14 interesting problem is presented when high input sample
15 rates (≥ 150 MHz) must be supported in FPGA technology. High-
16 performance multipliers are typically realized by
17 incorporating pipelining in the design. This naturally
18 introduces some latency in to the system. The location of
19 the predictor filter $P(z)$ requires a zero-latency design.
20 (It is possible that the predictor could be modified to
21 predict samples further ahead in the time series, but this
22 potential modification will not be dealt with in the limited
23 space available.) Instead of re-quantizing, filtering and
24 decimating, which would of course require a $\Sigma\Delta$ modulator
25 running at the input sample rate, this sequence of
26 operations is re-ordered to permit several slower modulators
27 to be used in parallel. The process is performed by first
28 decimating the signal, re-quantizing and then filtering. Now
29 the $\Sigma\Delta$ modulators operate at the reduced output sample rate.
30 This is depicted in Figure 15. To support arbitrary center
31 frequencies, and any arbitrary, but integer, down-sampling

RED-LINED VERSION

1 factor m , the bandpass decimation filter employs complex
2 weights. The filter weights are of course just the bandpass
3 modulated coefficients of a lowpass prototype filter
4 designed to support the bandwidth of the target signal.
5 Samples are collected from the A/D and alternated between
6 the two modulators. Both modulators are identical and use
7 the same predictor filter coefficients. The re-quantized
8 samples are processed by an $m:1$ complex polyphase filter to
9 produce the decimated signal. Several design options are
10 presented once the signal has been filtered and the sample
11 rate lowered. Figure 15 illustrates one possibility. Now
12 that the data rate has been reduced, the low rate signal is
13 easily shifted to baseband with a simple, and area
14 efficient, complex heterodyne. One multiplier and a single
15 digital frequency synthesizer could be time shared to
16 extract one or multiple channels.

17 It is interesting to investigate some of the changes
18 that are required to support the $\Sigma\Delta$ decimator. The center
19 frequency of the prediction filter should be designed to
20 predict samples in the required spectral region in
21 accordance with the output sample rate. For example,
22 consider $m=2$, and the required channel center frequency
23 located at 0.1 Hz, normalized with respect to the input
24 sample rate. The prediction filter should be designed with a
25 center frequency located at 0.2 Hz. In addition, the quality
26 of the prediction should be improved. With respect to the
27 output sample rate, the predictors are required to operate
28 over a wider fractional bandwidth. This implies more filter
29 coefficients in $P(z)$. The increase in complexity of this
30 component should be balanced against the savings that result
31 in the reduced complexity filter stage to confirm that a net

RED-LINED VERSION

1 savings in logic requirements is produced. To more clearly
2 demonstrate the approach, consider a 2:1 decimator, a
3 channel center frequency at 0.2 Hz and a 60 dB dynamic range
4 requirement.

5 Figure 16(a) shows the double-sided spectrum of the
6 input test signal. The input signal is commutated between $\Sigma\Delta_0$
7 and $\Sigma\Delta_1$ to produce the two low-precision sequences $\hat{x}_0(n)$ and
8 $\hat{x}_1(n)$. The respective spectrums of these two signals are shown
9 in Figures 16(b) and 16(c). The complex decimation filter
10 response is defined in Figure 16(d). After filtering, a
11 complex sample stream supported at the low output sample
12 rate is produced. This spectrum is shown in Figure 16(e).
13 Observe that the out-of-band components in the test signal
14 have been rejected by the specified amount and that the in-
15 band data meets the 60 dB dynamic range requirement. For
16 comparison, the signal spectrum resulting from applying the
17 processing stages in the order, re-quantize, filter and
18 decimate is shown in Figure 16(f). The interesting point to
19 note is that while the dual $\Sigma\Delta$ modulator approach satisfies
20 the system performance requirements, its out-of-band
21 performance is not quite as good as the response depicted in
22 Figure 16(f). The stopband performance of the dual modulator
23 architecture has degraded by approximately 6 dB. This can be
24 explained by noting that the shaping noise produced by each
25 modulator is essentially statistically independent. Since
26 there is no coupling between these two components prior to
27 filtering, complete phase cancellation of the modulator
28 noise cannot occur in the polyphase filter.

29

1 Discussion**SUMMARY**

2 Sigma-delta modulation ($\Sigma\Delta$) technology, together with
3 FPGA based signal processing hardware, is combined to
4 produce creative, high-performance and area-efficient
5 solutions to many signal processing problems.

6 In one embodiment of the invention, a conventional DC
7 canceler is modified to include a re-quantizer in the
8 feedback loop in the form of a $\Sigma\Delta$ modulator. In such
9 embodiments, the modulator can be a very simple 1st order
10 loop that can easily be implemented using an FPGA.

11 In another embodiment, a digital receiver employs a
12 processing chip, such as an FPGA, that includes a $\Sigma\Delta$
13 modulator to requantize oversampled control signals in the
14 digital receiver. This embodiment enables an FPGA to
15 generate analog signals to control low-bandwidth analog
16 functions using minimal additional hardware.

17 Still another embodiment of the invention is a wide-
18 bandwidth sigma-delta loop with a tunable center frequency.
19 In this embodiment, a fixed set of feedback weights from a
20 set of digital integrators defines a base-band filter with a
21 desirable noise transfer function. The filter is tuned to
22 arbitrary frequencies using a simple sub-processing element.
23 The low-pass to band-pass transformation for a sampled data
24 filter is achieved using an all-pass transfer function $G(z)$.
25 In this embodiment, tuning is trivially accomplished by
26 changing a multiplier in the all-pass network. The tuning
27 multipliers can be implemented as full multipliers in FPGA
28 hardware or as dynamically re-configured constant-
29 coefficient multipliers.

30 This summary does not purport to define the invention,
31 which is instead defined by the claims.

BRIEF DESCRIPTION OF THE FIGURES

Figure 1 depicts a generic FPGA architecture.

Figure 2 depicts a configurable logic block (CLB) of a programmable logic device.

Figure 3 shows the CLB matrix used in Xilinx Virtex FPGA's.

Figure 4 shows a logic block architecture employed in the Atmel AT40K FPGA.

Figure 5 shows a single-loop sigma-delta modulator.

Figure 6 shows a linearized discrete time model of a single-loop sigma-delta modulator.

Figure 7 depicts the noise transfer function of the modulator of Figure 6.

Figure 8 depicts a filter system that includes a sigma-delta modulator.

Figure 9 depicts a single loop error feedback sigma-delta modulator.

Figure 10 depicts a tunable sigma-delta modulator using a linear filter in the feedback path.

Figure 11A is a frequency domain plot of an input test signal for the modulator in Figure 10.

Figure 11B is a frequency domain plot of the input test signal of Figure 10.

Figure 11C is a frequency domain plot of the frequency response of a 160-tap filter $H(z)$.

Figure 11D is a frequency domain plot of a signal resulting from the application of filter $H(z)$ of Figure 11C to re-quantized test signal of Figure 11B.

Figure 12 is a frequency domain plot of band-limited white noise process $N_x(\omega)$.

1 Figure 13 depicts an FPGA FIR filter.

2 Figure 14 depicts a look-up table storing 16 scaled co-
3 efficient values for one tap of the FIR filter of Figure 13.

4 Figure 15 depicts a decimation filter that includes two
5 modulators operating in parallel.

6 Figure 16(a) shows the double-sided spectrum of an
7 input test signal for the decimators of Figure 15.

8 Figures 16(b) depicts the spectrum of sequence $\hat{x}_0(n)$ in
9 the decimator of Figure 15.

10 Figures 16(c) depicts the spectrum of sequence $\hat{x}_1(n)$ in
11 the decimator of Figure 15.

12 Figure 16(d) depicts the complex filter response for
13 the decimation filter of Figure 15.

14 Figure 16(e) depicts a complex sample stream produced
15 by the decimation filter of Figure 15.

16 Figure 16(f) depicts a signal spectrum resulting from
17 applying the filter processing stages of Figure 15 in a
18 different order.

19 Figure 17 depicts a fourth-order sigma-delta loop.

20 Figure 18 depicts the time and spectrum obtained by
21 using the loop of Figure 17 with a 4-bit quantizer.

22 Figure 19(a) is a block diagram of a digital filter
23 with the transfer function $G(z)$.

24 Figure 19(b) is a block diagram of a digital filter
25 with the transfer function for $G(z)$ and having a co-
26 efficient multiplier C.

27 Figure 20 depicts a tunable fourth order sigma-delta
28 loop.

29 Figure 21 depicts the time and spectrum obtained by
30 using the tunable loop shown in Figure 20.

31 Figure 22 is a graph depicting the relationship between

1 the coefficient c and the center frequency for the loop of
2 Figure 20.

3 Figure 23 depicts FPGA hardware adapted to implement a
4 loadable constant coefficient multiplier.

5 Figure 24 depicts a simple DC canceler.

6 Figure 25A is a spectral domain representation of a
7 biased signal presented to the DC canceler of Figure 24.

8 Figure 25B is the processed signal spectrum at $yq(n)$ in
9 the DC canceler of Figure 24.

10 Figure 25C depicts the signal spectrum at $yq(n)$ using
11 the quantizer $Q(\cdot)$.

12 Figure 25D demonstrates the operation of the DC
13 canceler of Figure 26 for 8-bit output data.

14 Figure 26 depicts a DC canceler with a re-quantizer
15 embedded in the feedback loop.

16 Figure 27 depicts a digital receiver with feedback
17 paths containing digital signals converted to analog control
18 signals for analog components.

19 Figure 28 depicts a digital receiver employing a sigma-
20 delta modulator to facilitate simplified the feedback
21 circuitry.

22 Figure 29 shows the time responses of the one-bit two
23 loop sigma-delta converter to a slowly varying control
24 signal and the reconstructed signal obtained from the dual-
25 RC filter.

26 Figure 30 shows the spectrum obtained from a 1-bit two-
27 loop modulator and the spectrum obtained from an unbuffered
28 RC-RC filter.

29

1 DETAILED DESCRIPTION

2 To provide a frame of reference for the $\Sigma\Delta$ decimator,
3 consider an implementation that does not pre-process the
4 input data, but just applies it directly to a polyphase
5 decimation filter. A complex filter processing real-valued
6 data consumes double the FPGA resources of a filter with
7 real weights. For $N=160$, 15344 CLBs are required. This
8 Figure is based on a cost of 40 CLBs for each KCM and 8 CLBs
9 for an add-delay component.

10 Now consider the logic accounting for the dual
11 modulator approach. The area cost $\Gamma(\widehat{\text{FIR}})$ for this filter is
12
13 Equation 22:

$$\Gamma(\widehat{\text{FIR}}) = 2\Gamma(\Sigma\Delta) + \Gamma(\widehat{\text{MUL}}) + \Gamma(\text{ACC}_z^{-1})$$

14
15 where $\Gamma\Sigma\Delta$ represents the logic requirements for one $\Sigma\Delta$
16 modulator, and $\Gamma(\widehat{\text{MUL}})$ is the logic needed for a reduced
17 precision multiplier. Using the filter specifications
18 defined earlier, and 18-tap error prediction filters,
19

$$\Gamma(\widehat{\text{FIR}}) = 2 \times 738 + 2 \times ((160 + 159) \times 8) = 6596.$$

20
21
22 Comparing the area requirements of the two options produces
23 the ratio
24

25 Equation 23:

$$\lambda = \frac{\Gamma(\widehat{\text{FIR}})}{\Gamma_{\text{FIR}}} = 6596/15344 \approx 43\%$$

26

1 So for this example, the re-quantization approach has
2 produced a realization that is significantly more area
3 efficient than a standard tapped-delay line implementation.
4

5 Center Frequency Tuning

6 For both the single-rate and multi-rate $\Sigma\Delta$ based
7 architectures, the center frequency is defined by the
8 coefficients in the predictor filter and the coefficients in
9 the primary filter. The constant coefficient multipliers can
10 be constructed using the FPGA function generators configured
11 as RAM elements. When the system center frequency is to be
12 changed, the system control hardware would update all of the
13 tables to reflect the new channel requirements. If only
14 several channel locations are anticipated, separate
15 configuration bit streams could be stored, and the FPGA(s)
16 re-configured as needed.
17

18 Bandpass $\Sigma\Delta$ s Using Allpass Networks

19 In an earlier section we discussed how to design a
20 predicting filter for the feedback loop of a standard sigma
21 delta modulator. The predicting filter increases the order
22 of the modulator so that the modified structure has
23 additional degrees of freedom relative to a single-delay
24 noise feedback loop. These extra degrees of freedom have
25 been used in two ways, first to broaden the bandwidth of the
26 loop's noise transfer function, and second to tune its
27 center frequency. The tuning process entailed an off line
28 solution of the Normal equations which, while not difficult,
29 does present a small delay and the need for a background
30 processor. We can define a sigma-delta loop with a
31 completely different architecture that offers the same

RED-LINED VERSION

1 flexibility, namely wider bandwidth and a tunable center
2 frequency that does not require this background task. In
3 this alternate architecture, a fixed set of feedback weights
4 from a set of digital integrators defines a base-band
5 prototype filter with a desirable NTF. The filter is tuned
6 to arbitrary frequencies by attaching to each delay element
7 z^{-1} , a simple sub-processing element that performs a base-
8 band to band-pass transformation of the prototype filter.
9 This processing element tunes the center frequency of its
10 host prototype with a single real and selectable scalar. The
11 structure of a fourth order prototype sigma-delta loop is
12 shown in Figure 17. The time and spectrum obtained by using
13 the loop with a 4-bit quantizer is shown in Figure 18. In
14 this structure the digital integrator poles are located on
15 the unit circle at DC. The local feedback (a1 and a2)
16 separates the poles by sliding them along the unit circle,
17 and the global feedback (b1, b2, b3 and b4) places these
18 poles in the feedback path of the quantizer so they become
19 noise transfer function zeros. These zeros are positioned to
20 form an equal-ripple stop band for the NTF. The coefficients
21 selected to match the NTF pole-zero locations to an elliptic
22 high pass filter. The single sided bandwidth of this fourth
23 order loop is approximately 4% of the input sample rate.

24 The low-pass to band-pass transformation for a sampled
25 data filter is achieved by substituting an all-pass transfer
26 function $G(z)$ for the all-pass transfer function z^{-1} . This
27 transformation is shown in equation 24.

28

29 Equation 24:

$$z^{-1} \longrightarrow -z^{-1} \left(\frac{1-cz}{z-c} \right)$$

1

2

3 A block diagram of a digital filter with the transfer
4 function for $G(z)$ is shown in Figure 19. Examining the left
5 hand block diagram, we find the transfer function from $x(n)$
6 to $y(n)$ is the all-pass network $-(1-cz)/(z-c)$, while the
7 transfer function from $x(n)$ to $v(n)$ is $-(1/z)(1-cz)/(z-c)$.
8 When we absorb the external negative sign change in the
9 internal adders of the filter we obtain the simple right-
10 hand side version of the desired transfer function $G(z)$.

11 After the block diagram substitution has been made, we
12 obtain Figure 20, the tunable version of the low-pass
13 prototype. The basic structure of the prototype remains the
14 same when we replace the delay with the tunable all-pass
15 network. The order of the filter is doubled by the
16 substitution since each delay is replaced by a second order
17 sub-filter. Tuning is trivially accomplished by changing the
18 c multiplier of the all-pass network. The tuned version of
19 the system reverts back to the prototype response if we set
20 c to 1.

21 Figure 21 presents the time and spectrum obtained by
22 using the tunable loop with a 4-bit quantizer shown in
23 Figure 20. The single sided bandwidth of the prototype
24 filter is distributed to the positive and negative spectral
25 bands of the tuned filter. Thus the two-sided bandwidth of
26 each spectral band is approximately 4% of the input sample
27 rate.

1 We now estimate the computational workload required to
2 operate the prototype and tunable filter. The prototype
3 filter has six coefficients to form the 4-poles and the 4-
4 zeros of the transfer function. The two a_k $k=0,1$ coefficients
5 determine the four zero locations. These are small
6 coefficients and can be set to simple binary scalars. The
7 values computed for this filter for a_1 and a_2 were 0.0594 and
8 0.0110. These can be approximated by $1/16$ and $1/128$, which
9 lead to no significant shift of the spectral zeros in the
10 NTF. These simple multiplications are virtually free in the
11 FPGA hardware since they are implemented with suitable
12 wiring. The four coefficients b_k $k=0,\dots,3$ are 1.000, 0.6311,
13 0.1916, and 0.0283 respectively were replaced with
14 coefficients containing one or two binary symbols to obtain
15 values 1.000 , $1/2+1/8$ (0.625), $1/8+1/16$ (0.1875) and $1/32$
16 (0.03125). When the sigma-delta loop ran with these
17 coefficients there was no discernable change in bandwidth or
18 attenuation level of the loop. The loop operates equally as
19 well in the tuning mode and the non-tuning mode with the
20 approximate coefficients listed above. Thus the only real
21 multiplies in the tunable sigma-delta loop are the c
22 coefficients of the all-pass networks. These networks are
23 unconditionally stable and always exhibit all-pass behavior
24 even in the presence of finite arithmetic and finite
25 coefficients. This is because the same coefficient forms the
26 numerator and the denominator. Errors in approximating the
27 coefficients for c simply result in a frequency shift of the
28 filter's tuned center. The c coefficient is determined from
29 the cosine of the center frequency (in radians/sample). The
30 curve for this relationship is shown in Figure 22. Also
31 shown is an error due to approximating c by $c+\delta c$. The

1 question is, what is the change in center frequency θ , from
2 $\theta + \delta\theta$ due to the approximation of c ? We can see that the slope
3 at the operating point on the cosine curve is $-\sin\theta$ so that
4 $\delta c / \delta\theta \approx -\sin(\theta)$ so that $\delta c \approx -\delta\theta \sin(\theta)$ is the required
5 precision to maintain a specified error. We note that tuning
6 sensitivity is most severe for small frequencies where $\sin(\theta)$
7 is near zero. The tolerance term, $\delta\theta \sin(\theta)$, is quadratic for
8 small frequencies, but the lowest frequency that can be
9 tuned by the loop is half the NTF pass-band bandwidth. For
10 the fourth order system described here, this bandwidth is 4%
11 of the sample rate, so the half-bandwidth angle is 2%, or
12 0.126 radians. To assure that the frequency to which the
13 loop is tuned has an error smaller than 1% of center
14 frequency, $\delta c < \delta\theta \sin(\theta) \Rightarrow \delta c < (0.126/100)(0.126) = 0.0002$,
15 which corresponds to a 14 bit coefficient. An error of less
16 than 10% center frequency can be achieved with 10 bit
17 coefficients.

18 The tuning multipliers could be implemented as full
19 multipliers in the FPGA hardware or as dynamically re-
20 configured KCMs, or KDCM, as shown in Figure 23. The later
21 approach conserves FPGA resources at the expense of
22 introducing a start-up penalty each time the center
23 frequency is changed. The start-up period is the
24 initialization time of the KCM LUT. When a new center
25 frequency is desired, the tuning constant is presented to
26 the k input of the KDCM and the load signal LD is asserted.
27 This starts the initialization engine, which requires 16
28 clock cycles to initialize 16 locations in the multiplier
29 LUT. The initialization engine relies on the automatic shift
30 mode of the Virtex LUTs. In this mode of operation a LUT's

RED-LINED VERSION

1 register contents are passed from one cell to the next cell
2 on each clock tick. This avoids the requirement for a
3 separate address generator and multiplexor in the
4 initialization hardware. Observe from Figure 23 that the
5 initialization engine only introduces a small amount of
6 additional hardware over that of a static KCM.

7 There is approximately a factor of 4 difference in the
8 area of a KDCM and full multiplier.

9

10 $\Sigma\Delta$ DC Canceler

11 Unwanted DC components can be introduced into a DSP
12 datapath at several places. It may be presented to the
13 system via an un-trimmed offset in the analog-to-digital
14 conversion pre-processing circuit, or may be attributed to
15 bias in the A/D converter itself. Even if the sampled input
16 signal has a zero mean, DC content can be introduced through
17 arithmetic truncation processes in the fixed-point datapath.
18 For example, in a multi-stage multi-rate filter, the
19 intermediate filter output samples may be quantized between
20 stages in order to compensate for the filter processing gain
21 and thereby keep the word-length requirements manageable.
22 The introduced DC bias can impact the dynamic range
23 performance of a system and potentially increase the error
24 rate in a digital receiver application.

25 In a fixed-point datapath, the bias can cause
26 unnecessary saturation events that would not occur if the DC
27 was not present in the system.

28 In a digital communication receiver employing M-ary QAM
29 modulation, the DC bias can interfere with the symbol
30 decision process, so causing incorrect decoding and
31 therefore increasing the bit error rate.

1 In some cases the introduced bias can be ignored and is
2 of no concern. However, for other applications it is
3 desirable to remove the DC component.

4 One solution to removing the unwanted DC level is to
5 employ a DC canceler.

6 A simple canceler is shown in Figure 24. It is easy to
7 show that the transfer function of the network is

8

9 ~~Figure~~ **Equation 25:**

$$H(z) = \frac{z-1}{z-(1-z)}$$

10

11

12 The cancellation is due to the transfer function zero at 0
13 Hz. The pole at $1-\mu$ controls the system bandwidth, and hence
14 the system transient response. The location of the zero at
15 $z=1$ removes the DC component in the signal, but there are
16 some problems with a practical implementation of this
17 circuit.

18 Figure 25A is a spectral domain representation of a
19 biased signal presented to the DC canceler. Figure 25B is
20 the processed signal spectrum at $y_q(n)$ in Figure 24. We
21 observe that the DC content in the input signal has been
22 completely removed. However, in the process of running the
23 canceling loop the network processing gain has caused a
24 dynamic range expansion. So although the sample stream $y_q(n)$
25 is a zero mean process, it requires a larger number of bits
26 to represent each sample than is desirable. The only option
27 with the circuit is to re-quantize $y_q(n)$ to produce $y(n)$
28 using the quantizer $Q(\cdot)$. The effect of this operation is
29 shown in Figure 25C, which demonstrates, not surprisingly,

RED-LINED VERSION

1 that after an 8-bit quantizer, the signal now has a DC
2 component and we are almost back to where we started. How
3 can the canceler be re-organized to avoid this
4 implementation pitfall? One option is to embed the re-
5 quantizer in the feedback loop in the form of a $\Sigma\Delta$ modulator
6 as shown in Figure 26. The modulator can be a very simple
7 1st order loop such as the error feedback $\Sigma\Delta$ modulator shown
8 in Figure 9. Figure 25D demonstrates the operation of the
9 circuit for 8-bit output data. Observe from the Figure that
10 the DC has been removed from the signal while employing the
11 same 8-bit output sample precision that was used in Figure
12 24. The simple $\Sigma\Delta$ M employed in the canceler is easily
13 implemented in an FPGA.

14

15 Simplify Digital Receiver Control Loops Using $\Sigma\Delta$ Modulators

16 In earlier sections we recognized that when a sampled
17 data input signal has a bandwidth that is a small fraction
18 of its sample rate the sample components from this
19 restricted bandwidth are highly correlated. We took
20 advantage of that correlation to use a digital sigma-delta
21 modulator to requantize the signal to a reduced number of
22 bits. The sigma-delta modulator encodes the input signal
23 with a reduced number of bits while preserving full input
24 precision over the signal bandwidth by placing the increased
25 noise due to requantization in out-of-band spectral
26 positions that are already scheduled to be rejected by
27 subsequent DSP processing. The purpose of this
28 requantization is to allow the subsequent DSP processing to
29 be performed with reduced arithmetic resource requirements
30 since the desired data is now represented by a smaller
31 number of bits.

1 A similar remodulation of data samples can be
2 employed for signals generated within a DSP process when the
3 bandwidth of the signals are small compared to the sample
4 rate of the process. A common example of this circumstance
5 is the generation of control signals used in feedback paths
6 of a digital receiver. These control signals include a gain
7 control signal for a voltage controlled amplifier in an
8 automatic gain control (AGC) loop and VCO (voltage
9 controlled oscillator) control signals in carrier recovery
10 and timing recovery loops. A block diagram of a receiver
11 with these specific controls signals is shown in Figure 27.
12 The control signals are generated from processes operating
13 at a sample rate appropriate to the input signal bandwidth.
14 The bandwidth of control loops in a receiver are usually a
15 very small fraction of the signal bandwidth, which means
16 that the control signals are very heavily oversampled. As a
17 typical example, in a cable TV modem, the input bandwidth is
18 6 MHz, the processing sample rate is 20 MHz, and the loop
19 bandwidth may be 50 kHz. For this example, the ratio of
20 sample rate to bandwidth is 400-to-1.

21 As seen in Figure 27, the process of delivering these
22 oversampled control signals to their respective control
23 points entails the transfer of 16 bit words to external
24 control registers, requiring appropriate busses, addressing,
25 and enable lines as well as the operation of 16-bit digital-
26 to-analog converters (DACs).

27 We can use a sigma-delta modulator to requantize the
28 16-bit oversampled control signals in the digital receiver
29 prior to passing them out of the processing chip. The sigma-
30 delta can preserve the required dynamic range over the
31 signal's restricted bandwidth with a one-bit output. As

1 suggested in Figure 28, the transfer of a single bit to
2 control the analog components is a significantly less
3 difficult task than the original. We no longer require
4 registers to accept the transfer, the busses to deliver the
5 bits, or the DAC to convert the digital data to the analog
6 levels the data represents. All that is needed a simple
7 filter (and likely an analog amplifier to satisfy drive
8 level and offset requirements). Experience shows that a 1-
9 bit, one-loop sigma-delta modulator could achieve 80 dB
10 dynamic range and requires a single RC filter to reconstruct
11 the analog signal. A two-loop sigma-delta modulator is
12 required to achieve 16-bit precision for which a double RC
13 filter is required to reconstruct the analog output signal.
14 Figure 29 shows the time response of the one-bit two loop
15 sigma-delta converter to a slowly varying control signal and
16 the reconstructed signal obtained from the dual-RC filter.
17 Figure 30 shows the spectrum obtained from a 1-bit two-loop
18 modulator and the spectrum obtained from an unbuffered RC-RC
19 filter.

20 This example has shown how with minimal additional
21 hardware, an FPGA can generate analog control signals to
22 control low-bandwidth analog functions in a system.
23 An observation worthy of note, is that the audio engineering
24 community has recognized the advantage offered by this
25 option of requantizing a 16-bit oversampled data stream to
26 1-bit data stream. In that community, the output signal is
27 intentionally upsampled by a factor of 64 and then
28 requantized to 1-bit in a process called a MASH converter.
29 Nearly all CD players use the MASH converter to deliver
30 analog audio signals.

31

1 What Have We Gained?

2 What has been achieved by expressing our signal
3 processing problems in terms of $\Sigma\Delta$ techniques?

4 The paper has demonstrated some $\Sigma\Delta$ techniques for the
5 compact implementation of certain types of filter and
6 control applications using FPGAs. This optimization can be
7 used in several ways to bring economic benefits to a
8 commercial design. By exploiting $\Sigma\Delta$ filter processes, a
9 given processing load may be realizable in a lower-density,
10 and hence less expensive, FPGA than is possible without
11 access to these techniques. An alternative would be to
12 perform more processing using the same hardware. For
13 example, processing multiple channels in a communication
14 system.

15 In addition to FPGA area trade-offs, the $\Sigma\Delta$ methods
16 can result in reduced power consumption in a design. Power P
17 may be expressed as

18

19 **Equation 26:**

$$P = CV^2 f_{clk}$$

20

21 where C is capacitance, V is voltage and f is the system
22 clock frequency. By reducing the silicon area requirements
23 of a filter, we can simultaneously reduce the power
24 consumption of the design. For the examples considered
25 earlier, logic resource savings of greater than 50% were
26 demonstrated. The savings is proportional to increased
27 efficiency in the system power budget, and this of course is
28 very important for mobile applications.

1 The $\Sigma\Delta$ AGC, timing and carrier recovery control loop
2 designs are also important examples in a industrial context.
3 The examples illustrated how the component count in a mixed
4 analog/digital system can be reduced. In fact, not only is
5 the component count reduced, but printed circuit board area
6 is minimized. This results in more reliable and physically
7 smaller implementations. The reduced component count also
8 results in reduced power consumption. In addition, since the
9 control loops no longer require wide output buses from the
10 FPGA to multi-bit DACs that generate analog control
11 voltages, power consumption is decreased because fewer FPGA
12 I/O pads are being driven.

13

14 References

15 The subject matter of this application is excerpted
16 from and article entitled "FPGA Signal Processing Using
17 Sigma-Delta Modulation," C. H. Dick and F. J. Harris, IEEE
18 Signal Processing Magazine (January 2000), which is
19 incorporated herein by reference. The following documents
20 may also be of interest, and are also incorporated by
21 reference.

22 [1] Atmel, AT40K/05/10/20/40 Data Sheet, 1999.

23 [2] J. A. C. Bingham, The Theory and Practice of Modem
24 Design, John Wiley & Sons, New York, 1998.

25 [3] J. C. Candy and G. C. Temes, "Oversampling Methods for
26 A/D and D/A Conversion" in Oversampling Delta Sigma
27 Data Converters - Theory Design and Simulation, IEEE
28 Press, New York, 1992.

29 [4] M. E. Frerking, Digital Signal Processing in
30 Communication Systems, Van Nostrand Reinhold, New York,
31 1994.

RED-LINED VERSION

- 1 [5] S. Haykin, "Modern Filters", Macmillan Publishing Co.,
2 New York, 1990.
- 3 [6] H. Meyr, M. Moeneclaey and S. A. Fechtal, Digital
4 Communication Receivers, John Wiley & Sons Inc., New
5 York, 1998.
- 6 [7] S. R. Norsworthy, R. Schreier and G. C. Temes, \Delta-
7 Sigma Data Converters", IEEE Press, Piscataway, NJ,
8 1997.
- 9 [8] D. A. Patterson and J. L. Hennessy, Computer
10 Architecture: A Quantitative Approach, Morgan Kaufmann
11 Publishers Inc., California, 1990.
- 12 [9] A. Peled and B. Liu, "A New Hardware Realization of
13 Digital Filters", IEEE Trans. on Acoust., Speech,
14 Signal Processing, vol. 22, pp. 456-462, Dec. 1974.
- 15 [10] J. G. Proakis, and D. G. Manolakis, Digital Signal
16 Processing Principles, Algorithms and Applications
17 Second Edition, Maxwell Macmillan International, New
18 York, 1992.
- 19 [11] S. A. White, "Applications of Distributed Arithmetic to
20 Digital Signal Processing", IEEE ASSP Magazine, Vol.
21 6(3), pp. 4-19, July 1989.
- 22 [12] Xilinx Inc., The Programmable Logic Data Book, 1999.

1 CLAIMS

2 **What is claimed is:**

3

4 1. A DC canceler circuit comprising:

5 a. a canceler input terminal adapted to receive a
6 series of data input samples $x(n)$;

7 b. a canceler output terminal adapted to provide a
8 series of data output samples $y(n)$;

9 c. a feedback path having:

10 i. a feedback-path input terminal connected to
11 the canceler output terminal;

12 ii. a feedback-path output terminal connected to
13 the canceler input terminal; and

14 iii. a sigma-delta modulator **having a sigma-delta**
15 **input terminal connected to the** ~~connected~~
16 ~~between the~~ feedback-path input terminal and a
17 **sigma-delta output terminal connected to the**
18 feedback-path output terminal.

19

20 2. The canceler circuit of claim 1, further comprising a
21 subtractor having a first subtractor input node
22 connected to the canceler input terminal and a second
23 subtractor input node connected to the sigma-delta
24 output terminal.

25

26 3. The canceler circuit of claim 1, further comprising a
27 unit delay element having a delay-element input
28 terminal connected to the feedback path input terminal
29 and a delay-element output terminal connected to the
30 sigma-delta input terminal.

31

- 1 4. The canceler circuit of claim 3, further comprising an
2 adder having a first adder input terminal connected to
3 the delay-element output terminal, a second adder input
4 terminal connected to the feedback path input terminal,
5 and an adder output terminal connected to the delay-
6 element input terminal.
7
- 8 5. The canceler circuit of claim 4, wherein the adder
9 connects to the feedback path input terminal via a
10 multiplier.
11
- 12 6. A receiver comprising:
13 a. a processing chip configured to include;
14 i. a data input port;
15 ii. a data output port;
16 iii. sigma-delta modulator connected to the data
17 input port and having a control-signal output
18 port; and
19 b. a feedback path connected between the control-
20 signal output port and the data input port.
21
- 22 7. The receiver of claim 6, wherein the feedback path
23 includes:
24 a. an analog component having a filter input
25 terminal; and
26 b. an analog filter connected between the control-
27 signal output port and the filter input terminal.
28
- 29 8. The receiver of claim 7, wherein the analog component
30 includes an automatic gain control circuit.
31

- 1 9. The receiver of claim 7, wherein the analog component
2 includes a voltage-controlled oscillator.
3
- 4 10. The receiver of claim 6, wherein the processing chip is
5 programmable logic device.
6
- 7 11. The receiver of claim 10, wherein the programmable
8 logic device is a field programmable gate array.
9
- 10 12. A sigma-delta loop having a tunable center frequency,
11 the loop comprising:
12 a. a data input terminal adapted to receive data
13 $x(n)$;
14 b. a tunable all-pass network having an all-pass
15 network input terminal connected to the data input
16 terminal and an all-pass network output terminal;
17 c. a global feedback network connected between the
18 all-pass network output terminal and the all-pass
19 network input terminal; and
20 d. a local feedback network connected between the
21 all-pass network output terminal and the all-pass
22 network input terminal.
23
- 24 13. The sigma-delta loop of claim 12, further comprising a
25 second tunable all-pass network having a second all-
26 pass network input terminal, connected to the first-
27 mentioned all-pass network output terminal, and a
28 second all-pass network output terminal.
29
- 30 14. The sigma-delta loop of claim 12, wherein the global
31 feedback network comprises:

- 1 a. a first co-efficient multiplier connected between
- 2 the first-mentioned all-pass network output
- 3 terminal and the first-mentioned all-pass network
- 4 input terminal; and
- 5 b. a second co-efficient multiplier connected between
- 6 the second all-pass network output terminal and
- 7 the first-mentioned all-pass network input
- 8 terminal.
- 9
- 10 15. The sigma-delta loop of claim 14, further comprising a
- 11 quantizer having a quantizer input terminal connected
- 12 to the global feedback network and a quantizer output
- 13 terminal connected to the first-mentioned all-pass
- 14 network input terminal.
- 15
- 16 16. A tunable sigma-delta loop comprising:
- 17 a. a data input terminal adapted to receive data
- 18 $x(n)$;
- 19 b. a first subtractor having a first input terminal,
- 20 a second input terminal, and an output terminal;
- 21 c. a second subtractor having a first input terminal
- 22 connected to the output terminal of the first
- 23 subtractor, a second input terminal, and an output
- 24 terminal;
- 25 d. a first adder having a first input terminal
- 26 connected to the output terminal of the second
- 27 adder, a second input terminal, and an output
- 28 terminal;
- 29 e. a tunable all-pass network having an all-pass
- 30 network input terminal connected to the output
- 31 terminal of the first adder and an all-pass

- 1 network output terminal connected to the second
2 input terminal of the first adder;
- 3 f. a local feedback network having a local-feedback
4 input terminal connected to the all-pass network
5 output terminal and a local-feedback output
6 terminal connected to the second input terminal of
7 the of the second subtractor;
- 8 g. a global feedback network having a global-feedback
9 input terminal connected to the all-pass network
10 output terminal and a global-feedback output
11 terminal; and
- 12 h. a quantizer having a quantizer input terminal
13 connected to the global-feedback output terminal
14 and a quantizer output terminal connected to the
15 second input terminal of the first subtractor.
- 16
- 17 17. The loop of claim 16, further comprising:
- 18 a. a second adder having a first adder input terminal
19 connected to the first-mentioned all-pass network
20 output terminal, a second adder input terminal,
21 and an adder output terminal connected to the
22 local-feedback input terminal;
- 23 b. a second tunable all-pass network having an all-
24 pass network input terminal connected to the
25 output terminal of the second adder and a all-pass
26 network output terminal connected to the second
27 input terminal of the second adder;
- 28 c. a second global feedback network having a global-
29 feedback input terminal connected to the all-pass
30 network output terminal of the second all-bass
31 network and a global-feedback output terminal;

RED-LINED VERSION

- 1 d. a third adder having a first input terminal
2 connected to the global-feedback output terminal
3 of the first-mentioned global feedback network, a
4 second input terminal connected to the global-
5 feedback output terminal of the second global
6 feedback network, and an output terminal connected
7 to the quantizer input terminal.
8

1 TUNABLE NARROW-BAND FILTER INCLUDING
2 SIGMA-DELTA MODULATOR

3
4 Christopher H. Dick
5 Frederic J. Harris
6

7 **ABSTRACT OF THE DISCLOSURE**

8 Sigma-delta modulation ($\Sigma\Delta$) techniques provide a range
9 of opportunities in a signal processing system for both
10 increasing performance and datapath optimization along the
11 silicon-area axis in the design space. $\Sigma\Delta$ technology,
12 together with FPGA based signal processing hardware, can be
13 combined to produce creative, high-performance and area-
14 efficient solutions to many signal processing problems.